Rwordseg使用说明

Jian Li*

1 简介及安装

1.1 简介

Rwordseg 是一个R环境下的中文分词工具，使用rJava调用Java分词工具Ansj。

Ansj 也是一个开源的Java 中文分词工具，基于中科院的ictclas 中文分词算法，采用隐马尔科夫模型（Hidden Markov Model, HMM）。作者孙健重写了一个Java版本，并且全部开源，使得Ansi 可用于人名识别、地名识别、组织机构名识别、多级词性标注、关键词提取、指纹提取等领域，支持行业词典、用户自定义词典。

详细信息可以参考作者孙健的专访（http://blog.csdn.net/blogdevteam/article/details/8148451）以及项目的Github 地址（https://github.com/ansjsun/ansj_seg）。

当前版本的Rwordseg包完全引用了Ansj 包，只是简单提供了R的接口，并根据R中处理文本的习惯进行了调整，在此对原作者孙健表示强烈的敬意！之所以没有命名为ransj，是因为以后可能还会纳入其他的分词工具或者自己开发一些新的功能。

1.2 安装

当前版本在Rforge（https://r-forge.r-project.org/R/?group_id=1054）进行维护，可以直接安装最新版。如果是老版的R，使用“source”的方式进行安装。

```
> install.packages("rJava")
> install.packages("Rwordseg", repos="http://R-Forge.R-project.org")
> #install.packages("Rwordseg", repos="http://R-Forge.R-project.org", type="source")
```

该包依赖于rJava包和Java环境，在安装之前需要确保JRE已经安装，并且正确地设置了环境变量。当前版本的R包在JRE 1.6.0_32下测试通过。如果不确定本机是否有JRE，可以搜索Oracle网站的JRE进行安装。

*电子邮件：rweibo@sina.com；新浪微博：lijian001；主页：http://jliblog.com/app/rwordseg

以Windows系统为例，假设 JRE 的安装目录为 D:\jdk1.6.0_32\jre，R的安装目录为 D:\R\R-3.0.2。需要将以下路径添加到 PATH 环境变量中：

- D:\jdk1.6.0_32\jre\bin
- D:\jdk1.6.0_32\jre\bin\server
- D:\jdk1.6.0_32\jre\bin\client
- D:\R\R-3.0.2\library\rJava\jri

注意查看 JRE 的安装目录中 jvm.dll 是位于 server 还是 client 子文件，确保该文件夹位于 PATH 环境变量即可。全部设置完毕后可以正常导入R包：

```
> library(Rwordseg)
```

```
Loading required package: rJava
# Version: 0.1-2
```

2 与其他分词包的对比

目前开源的分词工具有很多，但是集成到R中的不多。在这里我们对目前可用的工具进行简单的介绍，并针对一个例子比较分词的结果：

```
> teststring1 <- "小都督促织女将R语言学习到底！"
> teststring2 <- "小都是统计之都的昵称:)"
```

2.1 mmseg4j

目前最常用的R分词包可能是基于mmseg4j 的rmmseg4j ，因为其作者将其发布到了CRAN（`http://cran.r-project.org/web/packages/rmmseg4j/index.html`）。由于mmseg4j 之类的分词工具已经是之前好几代的技术，目前很少有人还在使用，所以对于R的用户造成了很大的困扰。很多用户之所以还在用这个包的原因是其能自定义词库。

我们首先使用基本的设置进行分词：

```
> require(rmmseg4j)
> mmseg4j(teststring1)
```

```
[1] "小 都 督促 织女 将 r 语言 学习 到底"
```

```
> mmseg4j(teststring2)
```

```
[1] "小 都是 统计 之都 的 昵称"
```

当前版本的rmmseg4j的默认词库中没有“促织”和“R语言”这两个词，因此我们将其添加到自定义词库（ %R_HOME%\library\rmmseg4j\userDic\words-rmmseg4j.dic ），再进行分词：

```
"小 都督 促织 女将 r 语言 学习 到底"
```

可以发现分词全部乱了。进行10000次分词，测试时间：

```
> system.time(for(i in 1:10000) mmseg4j(teststring1))
```

```
   user  system elapsed
 171.91    8.83  180.24
```

花费了3分钟左右。

2.2 imdict-chinese-analyzer

imdict-chinese-analyzer是imdict智能词典的智能中文分词模块，作者高小平，算法基于隐马尔科夫模型（Hidden Markov Model, HMM），是中国科学院计算技术研究所的ictclas中文分词程序的重新实现（基于Java），可以直接为lucene搜索引擎提供中文分词支持。

ictclas应该是目前分词准确度和效率最好的工具，不过是商业软件。imdict-chinese-analyzer是它的一个开源的JAVA版本，但是不能支持词性识别和自定义词库。

本Rwordseg包作者的主页（`http://jliblog.com/app/rsegword`）上有一个集成了imdict-chinese-analyzer的RsegWord包，另外rmmseg4j 的作者还提供了一个rsmartcn，基于smartcn，这是imdict 集成到lucene之后的结果，效果和imdict-chinese-analyzer一样。

我们使用RsegWord进行测试[1]：

```
> require(RsegWord)
> segWord(teststring1)
```

```
[1] "小"   "都"   "督促" "织女" "将"   "r"    "语言" "学习" "到底"
```

```
> segWord(teststring2)
```

```
[1] "小"   "都"   "是"   "统计" "之"   "都"   "的"   "昵"   "称"
```

从teststring2的例子可以看到其并未将“都是”识别成一个词，是一个进步，但是词库中没有“昵称”这个词，也没办法添加。

我们再看其进行10000次分词的时间：

[1]该包已经停止维护，这里的例子只是为了说明新版和旧版的不同。

```
> system.time(for(i in 1:10000) segWord(teststring1))
```

```
   user  system elapsed
  25.35    1.04   22.20
```

时间缩短到了25秒。

2.3 ansj

前面已经介绍了Ansj 也是基于中科院的ictclas 中文分词算法，但是作者用JAVA重写了整个系统，并且全部开放源代码。

基础的分词功能和imdict 差别不大，不过基础词典有更新，包含了“昵称”一词。

```
> require(Rwordseg)
> segmentCN(teststring1)
```

```
[1] "小"    "都"    "督促" "织女" "将"    "r"     "语言" "学习" "到底"
```

```
> segmentCN(teststring2)
```

```
[1] "小"    "都"    "是"    "统计" "之"    "都"    "的"    "昵称"
```

我们同样将“促织”和“R语言”这两个词加入词库：

```
> insertWords(c("促织", "R语言"))
> segmentCN(teststring1)
```

```
[1] "小"     "都"     "督促"   "织女"   "将"     "r语言" "学习"   "到底"
```

可以发现分词器并未受到新词的干扰，仍然保持了对句子的正确理解，同时中英文混合的词“R语言”也能正确地分出来。

测试10000次分词的时间：

```
> system.time(for(i in 1:10000) segmentCN(teststring1))
```

```
   user  system elapsed
   7.75    0.02    6.57
```

可以发现性能有了进一步地提升，只花费8秒不到。

3 分词操作

3.1 默认分词

默认参数下输入需要分词的句子，GBK或者UTF-8的编码都可以，注意需要在R的控制台中显示正常。

```
> segmentCN("结合成分子时")
```

```
[1] "结合" "成"   "分子" "时"
```

如果输入参数为字符向量，则返回列表：

```
> segmentCN(c("说的确实在理", "一次性交多少钱"))
```

```
[[1]]
[1] "说"   "的"   "确实" "在"   "理"

[[2]]
[1] "一次性" "交"     "多少"   "钱"
```

参数nosymbol表示是否只输出汉字、英文和数字，默认为TRUE，否则将会输入标点符号等。

```
> segmentCN("想渊明、《停云》诗就，此时风味。")
```

```
[1] "想"   "渊明" "停"   "云"   "诗"   "就"   "此时" "风味"
```

```
> segmentCN("想渊明、《停云》诗就，此时风味。", nosymbol = FALSE)
```

```
 [1] "想"   "渊明" "、"   "《"   "停"   "云"   "》"   "诗"   "就"   "，"   "此时" "风味" "。"
```

3.2 词性识别

参数nature可以设置是否输出词性，默认不输出，如果选择输出，那么返回的向量名为词性的标识：

```
> segmentCN("花一元钱买了一朵美丽的花", nature = TRUE)
```

```
    v     m     n     v    ul     m     a    uj     v
"花" "一元"  "钱"  "买"  "了" "一朵" "美丽"  "的"  "花"
```

不过目前的词性识别并不是真正意义上的智能词性识别，同一个词在语义上的词性还没有办法识别出来，结果仅作为参考。

关于词性符号的含义请参见表1。

表 1: 词性符号表

符号	词性	符号	词性	符号	词性
Ag	形语素	j	简称略语	r	代词
a	形容词	k	后接成分	s	处所词
ad	副形词	l	习用语	Tg	时语素
an	名形词	m	数词	t	时间词
b	区别词	Ng	名语素	u	助词
c	连词	n	名词	Vg	动语素
Dg	副语素	nr	人名	v	动词
d	副词	ns	地名	vd	副动词
e	叹词	nt	机构团体	vn	名动词
f	方位词	nz	其他专名	w	标点符号
g	语素	o	拟声词	x	非语素字
h	前接成分	p	介词	y	语气词
i	成语	q	量词	z	状态词

3.3 人名识别

isNameRecognition选项可以设置是否进行智能的人名识别。智能识别有时候会和自定义的词库冲突，因此默认的选型是不进行人名识别：

```
> getOption("isNameRecognition")
```

```
[1] FALSE
```

```
> segmentCN("梅野石是东昆仑盟主")
```

```
[1] "梅"    "野"    "石"    "是"    "东"    "昆仑" "盟主"
```

使用本包的函数segment.options 可以设置该选项：

```
> segment.options(isNameRecognition = TRUE)
> segmentCN("梅野石是东昆仑盟主")
```

```
[1] "梅野石" "是"      "东昆仑" "盟主"
```

```
> getOption("isNameRecognition")
```

```
[1] TRUE
```

需要注意的是，虽然我们可以使用getOption 函数来获取该选项的当前值，但是不能使用options 函数来设置选项，而是需要使用本包自带的segment.options 函数来设置，否则不会生效。

3.4 tm 格式的支持

segmentCN函数默认是输出向量和列表，并使用向量的name属性来表示词性，这是R中最常用的数据结构。但是由于tm 包已经成了R中事实的文本挖掘标准，因此常常会需要使用tm 中使用空格分隔的单字符串格式。

returnType 参数如果设置成“tm”，则表示输出tm 格式的字符串，需要注意的是，如果采用该方式，则无法输出词性。

isfast 参数将可以设定直接调用JAVA包进行最基础的分词，速度比较快，只能返回“tm”格式的文本，无法输出繁体字，也不能进行词性识别。如果对分词的效率要求比较高的话可以选用词参数。但是如果输入为文本，暂时不支持词参数，因此该参数的作用不是很大，仅供参考。

3.5 对文件的分词

输入参数strwords 除了可以是需要分词的字符向量以外，也可以是某个文本文件的路径。本包会自动判断是否为某个文件的路径，并自动识别字符编码，全部转换成UTF-8进行处理和输出。

如果输入文本路径，可以使用outfile 参数指定输出文件的名称和路径，默认是与输入文件同文件夹并在原文件名基础上添加“segment”。blocklines 表示每次读入的行数，默认是1000行，当输入文件比较大的时候可以根据电脑的性能来设置该参数。

```
> segmentCN("说岳全传_GBK.txt")
```

```
Output file:  D:\说岳全传_GBK.segment.txt
[1] TRUE
```

该文件包含大约43万个汉字，分词的时间差不多5秒：

```
> system.time(segmentCN("说岳全传_GBK.txt"))
```

```
Output file:  D:\说岳全传_GBK.segment.txt
   user  system elapsed
    5.1     0.0     5.1
```

4 词典管理

4.1 安装和卸载词典

该包支持安装新的词典，一次安装之后，每次重启R包事都会自动加载。目前支持普通格式的文本词典和Sogou的Secl格式细胞词典。

函数 installDict 用来安装新词典，参数 dictpath 表示词典的路径；参数 dictname 表示自定义的词典名称，需要输入英文单词；参数 dicttype 表示词典的类型，目前只支持

“text”和“scel”，分别表示普通文本格式和Sogou细胞词典，默认为“text”，如果后缀为其他格式（比如.scel），则自动当作该格式导入。参数 load 表示安装后是否自动加载到内存，默认是 TRUE，表示自动加载，如果词典较大的话需要花费一段时间，如果设置为 FALSE，则安装字典后不加载到内存，在下次启动R及Rwordseg包时加载。

普通文本格式的后缀名没有限制，ANSI和UTF-8的文件都可以支持，需要确保文件用编辑器打开时能正常显示中文。我们可以安装ansj项目中的自定义词库（`https://github.com/ansjsun/ansj_seg/blob/master/library/default.dic`）。安装前：

```
> listDict()
```

```
[1] Name Type Des  Path
<0 rows> (or 0-length row.names)
```

```
> segmentCN("湖北大鼓真是不错啊")
```

```
[1] "湖北" "大鼓" "真"   "是"   "不错" "啊"
```

安装新的词典：

```
> installDict("E:/default.dic")
```

```
427450 words were loaded! ... New dictionary 'ansj' was installed!
```

成功安装后新词已经被添加：

```
> segmentCN("湖北大鼓真是不错啊")
```

```
[1] "湖北大鼓" "真是"     "不错"     "啊"
```

查看已安装的词典：

```
> listDict()
```

```
  Name Type         Des                                          Path
1 ansj Ansj default.dic D:/R/R-3.0.2/library/Rwordseg/dict/ansj.dic
```

以上的例子是一个巨大的词库，平常的使用中并不建议一次添加一个巨大的词库，因为分词的模型本来就是可以对新词进行划分，词库太大的效果并不一定好，而且加载时比较浪费时间。这个例子只是展示如何加载这种格式的词典。

在这里强烈推荐Sogou的细胞词库（`http://pinyin.sogou.com/dict/`），包含数目非常巨大的分类词典，而且可以找到最新的流行词。建议在进行具体研究之前找到相关的专业词典进行安装，比如要研究金庸的小说，可以下载词典“金庸武功招式.scel”。仍然使用installDict 来安装。未安装前的分词效果：

```
> segmentCN("真武七截阵和天罡北斗阵哪个厉害")
```

```
[1] "真武" "七"   "截"   "阵"   "和"   "天罡" "北斗" "阵"   "哪个" "厉害"
```

安装新的词典：

```
> installDict("E:\\金庸武功招式.scel", dictname = "jinyong")
```

```
932 words were loaded! ... New dictionary 'jinyong' was installed!
```

成功安装后新词已经被添加：

```
> segmentCN("真武七截阵和天罡北斗阵哪个厉害")
```

```
[1] "真武七截阵" "和"          "天罡北斗阵" "哪个"        "厉害"
```

查看已安装的词典：

```
> listDict()
```

```
     Name Type                        Des                                                Path
1    ansj Ansj                default.dic    D:/R/R-3.0.2/library/Rwordseg/dict/ansj.dic
2 jinyong 读书 金庸小说中的武功和招式名称 D:/R/R-3.0.2/library/Rwordseg/dict/jinyong.dic
```

如果想恢复到原状，可以卸载某些词典，默认卸载所有：

```
> uninstallDict()
```

```
427450 words were removed! ... The dictionary 'ansj' was uninstalled!
932 words were removed! ... The dictionary 'jinyong' was uninstalled!
```

卸载后直接生效：

```
> require(Rwordseg)
> segmentCN("湖北大鼓真是不错啊")
```

```
[1] "湖北" "大鼓" "真"   "是"   "不错" "啊"
```

```
> listDict()
```

```
[1] Name Type Des  Path
<0 rows> (or 0-length row.names)
```

4.2 自定义文本词典

如果仅仅只是用户自己添加词汇，没有必要做一个词典进行安装，可以使用自定义词典的功能。默认的词典目录为%R_HOME%\library\Rwordseg\dict。可以在其中添加任意后缀为.dic的文本，里面可以输入自定义的词，每一行一个词，回车换行。可以直接写在example.dic 这个文件，或者参考该文件新建dic 文件。

修改后每次重启时都会导入用户自定义词典。如果想要立即生效，可以在修改后运行：

```
> loadDict()
```

4.3 手动添加和删除词汇

如果仅仅只是在内存中临时添加或者删除词汇，可以使用insertWords和deleteWords：

```
> segmentCN("画角声断谯门")
[1] "画角" "声"   "断"   "谯"   "门"
> insertWords("谯门")
> segmentCN("画角声断谯门")
[1] "画角" "声"   "断"   "谯门"
> deleteWords(c("画角", "谯门"))
> segmentCN("画角声断谯门")
[1] "画" "角" "声" "断" "谯" "门"
```

如果需要将添加或者删除的词汇记录下来，每次重启时都会自动添加或者删除这些词，可以将save 参数设置为TRUE，例如：

```
> insertWords("谯门", save = TRUE)
> deleteWords(c("画角", "谯门"), save = TRUE)
```

更详细的介绍请查看函数的帮助文档。

5 台灣繁體字的支持

從Rwordseg 0.1-1 開始，提供了台灣繁體字的支持。對於UTF-8、GBK、和BIG5 的編碼環境都能正常處理，並且添加了台灣常用術語的詞庫，例如：

```
> segmentCN("使用R軟體蒐集資料")
[1] "使用" "R"    "軟體" "蒐集" "資料"
```

對於所有的操作，簡體中文和繁體中文都可以使用相同的方法實現。同樣可以維護繁體字的詞典。例如字典文件夹中的%R_HOME%\library\Rwordseg\dict\tw.dic 文件，可以在該文件的基礎上補充新詞，或者按照相同的格式創建新的詞典。